Please **AMEND** the **CLAIMS** as follows:

Please **CANCEL** claims 1, 19-28, and 34-35.

1. (Cancelled)

2. (Cancelled)

3. (Currently Amended) The method as recited in claim 11 ~~1~~, wherein the loading step is performed simultaneous with the building step.

4. (Currently Amended) The method as recited in claim 11 ~~1~~, wherein building a chain is performed such that the one or more code modules can be modified without requiring recompilation of the one or more code modules.

5. (Currently Amended) The method as recited in claim 11 ~~1~~, wherein loading the one or more code modules is performed in a reverse order of the hierarchical order.

6. (Currently Amended) The method as recited in claim 11 ~~1~~, wherein determining one or more code modules to be executed comprises determining one or more code modules to be executed to complete configuration of a hardware interface of a router.

7. (Currently Amended) The method as recited in claim 11 ~~1~~, wherein determining one or more code modules to be executed comprises determining one or more code modules to be executed to configure a router.

8. (Cancelled)

9. (Cancelled)

10. (Currently Amended) A method of linking a set of code modules for execution,

comprising:

determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

ascertaining a hierarchical order in which the one or more code modules are to be executed;

loading the one or more code modules to be executed; and

building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

~~The method as recited in claim 1,~~ wherein building a chain connecting the one or more code modules comprises:

obtaining a first one of the one or more code modules, wherein the first one of the one or more code modules has previously been loaded;

determining whether the first one of the one or more code modules is to subsequently execute a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules, wherein the second one of the one or more code modules has previously been loaded;

when it is determined that the first one of the one or more code modules is to subsequently execute a second one of the one or more code modules, updating a branch table of the first one of the one or more code modules to identify an address at which the second one of the one or more code modules is loaded such that the reference to the second one of the one or more code modules in the branch table of the first one of the one or more code modules is associated with the address at which the second one of the one or more code modules is loaded.

11. (Currently Amended) A method of linking a set of code modules for execution,

comprising:

determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

ascertaining a hierarchical order in which the one or more code modules are to be executed;

loading the one or more code modules to be executed; and

building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

~~The method as recited in claim 1,~~ wherein building a chain connecting the one or more code modules comprises:

obtaining a first one of the one or more code modules;

determining whether the first one of the one or more code modules has an option of executing a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules;

when it is determined that the first one of the one or more code modules has an option of executing a second one of the one or more code modules, updating a branch table in the first one of the one or more code modules to identify an address at which second one of the one or more code modules is loaded such that the address of the second one of the one or more code modules is associated with the reference to the second one of the one or more code modules.

12. (Currently Amended) A method of linking a set of code modules for execution, comprising:

determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

ascertaining a hierarchical order in which the one or more code modules are to be executed;

loading the one or more code modules to be executed; and

building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

~~The method as recited in claim 1,~~ wherein building a chain connecting the one or more code modules comprises:

obtaining a first one of the one or more code modules;

determining whether the first one of the one or more code modules can subsequently execute a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules;

when it is determined that the first one of the one or more code modules can subsequently execute a second one of the one or more code modules, updating a branch table in the first one of the one or more code modules to identify an address at which the second one of the one or more code modules has been loaded such that the address of the second one of the one or more code modules is associated with the reference to the second one of the one or more code modules.

13. (Previously Amended) The method as recited in claim 12, wherein the branch table of the first one of the one or more code modules includes the reference to the second one of the one or more code modules prior to loading the code modules and includes the address of the second one of the one or more code modules after the code modules have been loaded.

14. (Previously Amended) The method as recited in claim 12, wherein updating a

branch table includes modifying an entry in the branch table such that a dummy address is replaced with the address of the second one of the one or more code modules.

15. (Previously Amended) The method as recited in claim 12, wherein when the first one of the one or more code modules is shared by two or more executable chains of code modules, associating the second one of the one or more code modules with one of the two or more executable chains such that the branch table of the first one of the one or more code modules includes at least two entries, each of the entries identifying one of the two or more executable chains, each of the entries including an address.

16. (Previously Amended) The method as recited in claim 15, wherein the second one of the one or more code modules is associated with one of the two or more executable chains when a parameter is associated with one of the two or more executable chains such that each of the entries further includes a parameter used to select one of the two or more executable chains.

17. (Previously Amended) The method as recited in claim 12, wherein updating the branch table further includes replacing a dummy address associated with the reference to the second one of the one or more code modules with the address of the second one of the one or more code modules.

18. (Currently Amended) The method as recited in claim 11 ~~1~~, further comprising associating one of the one or more code modules with a hardware interface to identify a starting point for execution upon occurrence of an interrupt.

19. (Cancelled)

20. (Cancelled)

21. (Cancelled)

22. (Cancelled)

23. (Cancelled)

24. (Cancelled)

25. (Cancelled)

26. (Cancelled)

27. (Cancelled)

28. (Cancelled)

29. (Previously Added) The method as recited in claim 11, wherein updating the branch table in the first one of the one or more code modules comprises:

updating a conditional branch or jump instruction identifying the second one of the one or more code modules to include the address of the second one of the one or more code modules.

30. (Previously Added) The method as recited in claim 29, wherein updating the conditional branch or jump instruction comprises:

replacing a dummy address in the conditional branch or jump instruction with the address of the second one of the one or more code modules.

31. (Previously Added) The method as recited in claim 29, wherein the conditional branch or jump statement identifying the second one of the one or more code modules is executed when the second one of the code modules in the chain identified in the conditional branch or jump statement is executed.

32. (Previously Added) The method as recited in claim 12, wherein updating a branch table in the first one of the one or more code modules to identify an address at which the second one of the one or more code modules has been loaded such that the address of the second one of the one or more code modules is associated with the reference to the second one of the one or more code modules comprises:

updating a conditional branch or jump instruction identifying the second one of the one or more code modules to include the address of the second one of the one or more code modules.

33. (Previously Added) The method as recited in claim 32, wherein the conditional branch or jump statement identifying the second one of the one or more code modules is executed when the second one of the code modules in the chain identified in the conditional branch or jump statement is executed.

34. (Cancelled)

35. (Cancelled)

36. (Previously Added) The method as recited in claim 10, wherein the loading step is performed simultaneous with the building step.

37. (Previously Added) The method as recited in claim 11, wherein the loading step is performed simultaneous with the building step.

38. (Previously Added) The method as recited in claim 12, wherein the loading step is performed simultaneous with the building step.

39. (Previously Added) The method as recited in claim 10, further comprising:
repeating the obtaining, determining and updating steps for each of the code modules.

40. (Previously Added) The method as recited in claim 11, further comprising:
repeating the obtaining, determining and updating steps for each of the code modules.

41. (Previously Added) The method as recited in claim 12, further comprising:
repeating the obtaining, determining and updating steps for each of the code modules.

Please A**DD** claims 42-50 as follows:

42. (New) A computer-readable medium storing thereon computer-readable instructions for linking a set of code modules for execution, comprising:

instructions for determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

instructions for ascertaining a hierarchical order in which the one or more code modules are to be executed;

instructions for loading the one or more code modules to be executed; and

instructions for building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

wherein the instructions for building a chain connecting the one or more code modules includes:

instructions for obtaining a first one of the one or more code modules, wherein the first one of the one or more code modules has previously been loaded;

instructions for determining whether the first one of the one or more code modules is to subsequently execute a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules, wherein the second one of the one or more code modules has previously been loaded; and

instructions for updating a branch table of the first one of the one or more code modules to identify an address at which the second one of the one or more code modules is loaded such that the reference to the second one of the one or more code modules in the branch table of the first one of the one or more code modules is associated with the address at which the second one of the one or more code modules is loaded when it is determined that the first one of the one or more code modules is

to subsequently execute a second one of the one or more code modules.

43. (New) An apparatus for linking a set of code modules for execution, comprising:

means for determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

means for ascertaining a hierarchical order in which the one or more code modules are to be executed;

means for loading the one or more code modules to be executed; and

means for building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

wherein the means for building a chain connecting the one or more code modules includes:

means for obtaining a first one of the one or more code modules, wherein the first one of the one or more code modules has previously been loaded;

means for determining whether the first one of the one or more code modules is to subsequently execute a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules, wherein the second one of the one or more code modules has previously been loaded; and

means for updating a branch table of the first one of the one or more code modules to identify an address at which the second one of the one or more code modules is loaded such that the reference to the second one of the one or more code modules in the branch table of the first one of the one or more code modules is associated with the address at which the second one of the one or more code modules is loaded when it is determined that the first one of the one or more code modules is to subsequently execute a second one of the one or more code modules.

44. (New) An apparatus for linking a set of code modules for execution, comprising:

a processor; and

a memory, at least one of the processor and the memory being adapted for:

determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

ascertaining a hierarchical order in which the one or more code modules are to be executed;

loading the one or more code modules to be executed; and

building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

wherein building a chain connecting the one or more code modules includes:

obtaining a first one of the one or more code modules, wherein the first one of the one or more code modules has previously been loaded;

determining whether the first one of the one or more code modules is to subsequently execute a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules, wherein the second one of the one or more code modules has previously been loaded; and

when it is determined that the first one of the one or more code modules is to subsequently execute a second one of the one or more code modules, updating a branch table of the first one of the one or more code modules to identify an address at which the second one of the one or more code modules is loaded such that the reference to the second one of the one or more code modules in the branch table of the first one of the one or more code modules is associated with the address at which the second one of the one or more code modules is loaded.

45. (New) A computer-readable medium storing thereon computer-readable instructions for linking a set of code modules for execution, comprising:

instructions for determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

instructions for ascertaining a hierarchical order in which the one or more code modules are to be executed;

instructions for loading the one or more code modules to be executed; and

instructions for building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

wherein the instructions for building a chain connecting the one or more code modules includes:

instructions for obtaining a first one of the one or more code modules;

instructions for determining whether the first one of the one or more code modules has an option of executing a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules; and

instructions for updating a branch table in the first one of the one or more code modules to identify an address at which second one of the one or more code modules is loaded such that the address of the second one of the one or more code modules is associated with the reference to the second one of the one or more code modules when it is determined that the first one of the one or more code modules has an option of executing a second one of the one or more code modules.

46. (New) An apparatus for linking a set of code modules for execution, comprising:

means for determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

means for ascertaining a hierarchical order in which the one or more code modules are

to be executed;

means for loading the one or more code modules to be executed; and

means for building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

wherein the means for building a chain connecting the one or more code modules includes:

means for obtaining a first one of the one or more code modules;

means for determining whether the first one of the one or more code modules has an option of executing a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules; and

means for updating a branch table in the first one of the one or more code modules to identify an address at which second one of the one or more code modules is loaded such that the address of the second one of the one or more code modules is associated with the reference to the second one of the one or more code modules when it is determined that the first one of the one or more code modules has an option of executing a second one of the one or more code modules.

47. (New) An apparatus for linking a set of code modules for execution, comprising:

a processor; and

a memory, at least one of the processor and the memory being adapted for:

determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

ascertaining a hierarchical order in which the one or more code modules are to be executed;

loading the one or more code modules to be executed; and

building a chain connecting the one or more code modules such that the one or more

code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

wherein building a chain connecting the one or more code modules includes:

obtaining a first one of the one or more code modules;

determining whether the first one of the one or more code modules has an option of executing a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules; and

when it is determined that the first one of the one or more code modules has an option of executing a second one of the one or more code modules, updating a branch table in the first one of the one or more code modules to identify an address at which second one of the one or more code modules is loaded such that the address of the second one of the one or more code modules is associated with the reference to the second one of the one or more code modules.

48. (New) A computer-readable medium storing thereon computer-readable instructions for linking a set of code modules for execution, comprising:

instructions for determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

instructions for ascertaining a hierarchical order in which the one or more code modules are to be executed;

instructions for loading the one or more code modules to be executed; and

instructions for building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next

one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

wherein the instructions for building a chain connecting the one or more code modules includes:

instructions for obtaining a first one of the one or more code modules;

instructions for determining whether the first one of the one or more code modules can subsequently execute a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules; and

instructions for updating a branch table in the first one of the one or more code modules to identify an address at which the second one of the one or more code modules has been loaded such that the address of the second one of the one or more code modules is associated with the reference to the second one of the one or more code modules when it is determined that the first one of the one or more code modules can subsequently execute a second one of the one or more code modules.

49. (New) An apparatus for linking a set of code modules for execution, comprising:

means for determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

means for ascertaining a hierarchical order in which the one or more code modules are to be executed;

means for loading the one or more code modules to be executed; and

means for building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

wherein the means for building a chain connecting the one or more code modules

includes:

means for obtaining a first one of the one or more code modules;

means for determining whether the first one of the one or more code modules can subsequently execute a second one of the one or more code modules upon completion of execution of the first one of the one or more code modules; and

means for updating a branch table in the first one of the one or more code modules to identify an address at which the second one of the one or more code modules has been loaded such that the address of the second one of the one or more code modules is associated with the reference to the second one of the one or more code modules when it is determined that the first one of the one or more code modules can subsequently execute a second one of the one or more code modules.

50. (New) An apparatus for linking a set of code modules for execution, comprising:

a processor; and

a memory, at least one of the processor and the memory being adapted for:

determining one or more code modules to be executed, wherein the one or more code modules are one or more DLLs;

ascertaining a hierarchical order in which the one or more code modules are to be executed;

loading the one or more code modules to be executed; and

building a chain connecting the one or more code modules such that the one or more code modules will automatically execute in the hierarchical order when a first one of the one or more code modules is executed, wherein each of the code modules responsible for calling a next one of the code modules in the chain includes a reference to the next one of the code modules in the chain, wherein an address in memory at which the next one of the code modules in the chain is loaded is associated with the reference to the next one of the code modules in the chain, wherein the chain does not include a main program and wherein building a chain enables the one or more code modules to execute without requiring a main program responsible for calling the one or more code modules;

wherein building a chain connecting the one or more code modules includes:

obtaining a first one of the one or more code modules;

determining whether the first one of the one or more code modules can subsequently execute a second one of the one or more code modules upon completion

of execution of the first one of the one or more code modules; and

when it is determined that the first one of the one or more code modules can subsequently execute a second one of the one or more code modules, updating a branch table in the first one of the one or more code modules to identify an address at which the second one of the one or more code modules has been loaded such that the address of the second one of the one or more code modules is associated with the reference to the second one of the one or more code modules.